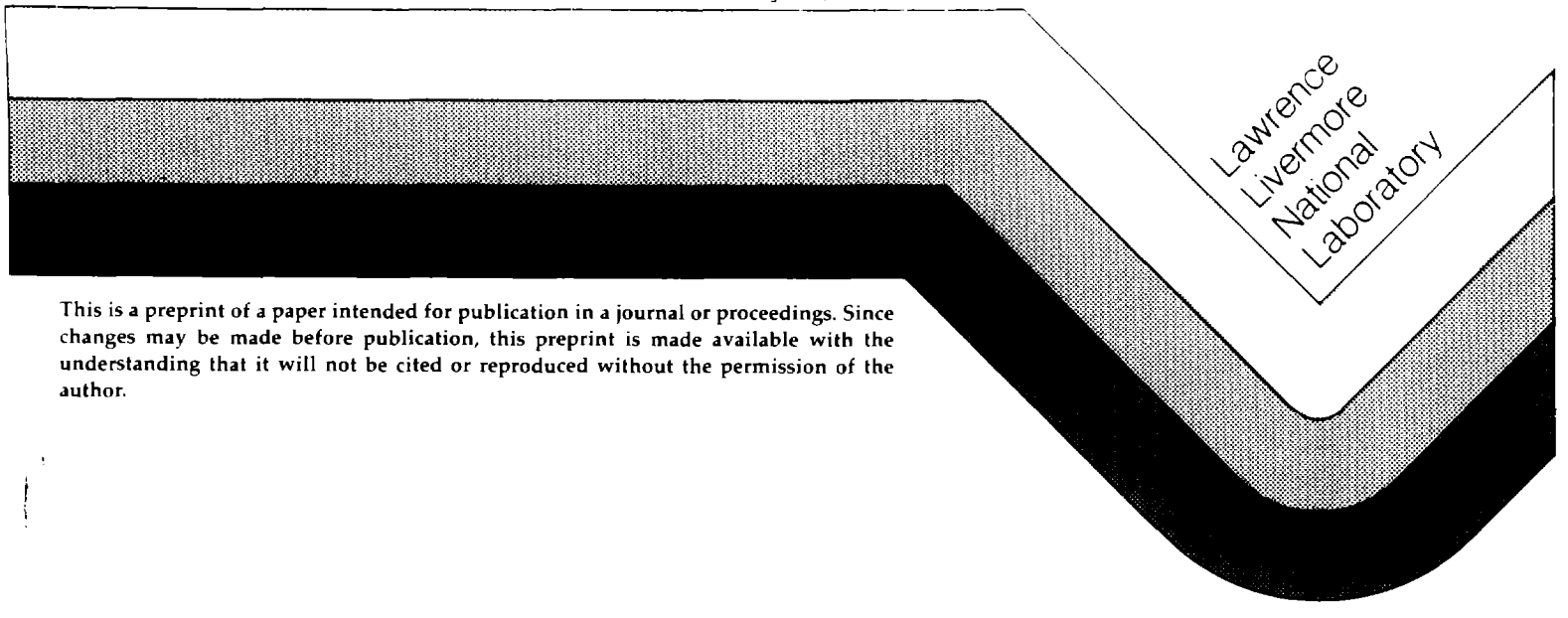
UCRL- 99088
PREPRINT

Parallel Issues Relating to Domain Decomposition Techniques for Solving PDE's

Ted Ferretta
Matt Nolan
Garry Rodrigue

This Paper Was Prepared For Submittal To
IFIP WG 2.5 Working Conference
Stanford University, 8/22-26/88

July 6, 1988



1. Domain Decomposition

In both loosely and tightly coupled multiprocessing systems, the primary breakdowns in efficiency are excessive communication between computer modules and idle processors. Thus, efficient numerical techniques for solving partial differential equations on multiprocessors are those that contain "substantial" numerical tasks having the property that they can be assigned to individual computer modules and they are mathematically independent as possible. The latter property quantifies the communication costs between the computer modules in that the mathematical dependencies between the numerical tasks is directly proportional to the numerical data traffic between the processors.

A methodology for determining numerical methods with the above properties can be developed by a decomposition of the spatial domain of the differential equation. Specifically, let us consider the numerical solution of the partial differential equation

$$F(x, u, t, D_t u, D_x u, D_x^2 u, \ldots) = 0, \tag{1.1}$$

where $x \in \Omega \subset R^n$ is a closed, bounded spatial domain and $u(x,t)$ satisfies given boundary and initial conditions. Then, at a given computational time t^*, the domain Ω is expressed as a union of subdomains (not necessarily disjoint)

$$\Omega = \bigcup_{i=1}^{k(t^*)} \Omega_i(t^*)$$

where $k(t^*) \geq 1$ is an algorithmically determined integer. This decomposition is then followed by defining on each subdomain $\Omega_i(t^*)$ a related differential equation

$$F_i(x, u_i, t, D_t u_i, D_x u_i, \ldots) = 0 \tag{1.2}$$

where $x \in \Omega_i(t^*)$ and u_i satisfies given boundary and initial conditions. The numerical task for each computer module of a multiprocessing system will be to solve one of the differential equations (1.2) and the communication costs will be the conveyance of properties of the numerical solution that are needed by the other processors so that they can proceed with the numerical solution of their differential equation.

In the case of steady-state problems we have $u_t = 0$ in (1.1) so that the above procedure does not have meaning. However, in this situation, the domain decomposition process takes the form of

an iteration where at each iterative step m, Ω is decomposed into $k(m)$ subdomains.

$$\Omega = \bigcup_{i=1}^{k(m)} \Omega_i(m)$$

and $k(m)$ differential equations on each of the subdomains are defined. Each processor then assumes the responsibility of solving one of the subproblem and then conveying pertinent information about the numerical solution to the other processors so that the next iteration can be carried out. To illustrate the preceding idea, we consider some domain decomposition processes for solving an elliptic partial differential equation that have been collectively called Schwarz Methods.

2. Schwarz Methods

For purposes of illustration, we consider the solution of the Dirichlet problem

$$F(u) = \frac{\partial}{\partial x} a(x,y) \frac{\partial u}{\partial x} + \frac{\partial}{\partial y} b(x,y) \frac{\partial u}{\partial y} = f(x,y),$$

$$(x,y) \varepsilon \Omega = [0,1] \times [0,1], \tag{2.1}$$

$$u\,|_\Gamma = \varphi(x,y) \quad , \quad \Gamma = \text{ boundary } (\Omega).$$

Here, the functions a, b, f, and φ are given. Let η_k be a positive integer and define the grid

$$G_k = \{(x_i, y_j) \ : \ X_i = i/\eta_k + 1 \ ; \ y_j = J/\eta_k + 1 \ ; \ i,j = 0,1,\ldots,\eta_k\} \tag{2.2}$$

Let $0 < p < \eta_k + 1$ be an integer and define the sequence of integers $\{r_i\}_{i=1}^p$, $\{\ell_i\}_{i=1}^p$ where

$$o = \ell_1 < \ell_2 < r_1 < \ell_3 < r_2 < \cdots < r_{p-1} < r_p = \eta_k + 1.$$

Let

$$\Omega_i = [x_{r_i}, x_{\ell_i}] \times [0,1] \ , \ 1 \le i \le p. \tag{2.3}$$

Then (2.3) defines a domain decomposition of Ω. Now, let $\{i, i_2, \ldots, i_p\}$ be a permutation of the set $\{1,2,\ldots,p\}$. Then for $k = 0,1,2,\ldots$ let

$$\begin{cases} F(u_{ij}^{(k)}) &= f_{ij} \text{ in } \Omega_{ij} \\ u_{ij}^{(k)} &= \varphi \text{ in } \partial(\Omega_{ij}) \bigcap \partial(\Omega) \\ u_{ij}^{(k)} &= u_{ij-1}^{(m_r)} \text{ in } \partial(\Omega_{ij}) \bigcap \partial(\Omega_{ij-1}) \\ u_{ij}^{(k)} &= u_{ij+1}^{(m\ell)} \text{ in } \partial(\Omega_{ij}) \bigcap \partial(\Omega_{ij+1}) \end{cases} \tag{2.4}$$

where $u_{ij}^{(0)}, j = 1, 2, \ldots, p$ are initial guesses. For example, we could take $m_r = k$ and $m_\ell = k - 1$ and

i) $\{i_1, i_2, \ldots, i_p\} = \{1, 2, \ldots, p\}$

or

ii) $\{i_1, i_2, \ldots, i_p\} = \{1, 3, 5, \ldots, 2, 4, \ldots\}$

3. Numerical Schwarz Methods

Numerical implementation of the Schwarz methods first involve a discrete approximation of the equations in (2.4) in a grid G_k and then for $k = 0, 1, 2, \ldots$, we obtain grid approximations $\bar{u}_{ij}^{(k)}$ to $u_{ij}^{(k)}$ on Ω_{ij}. How this is done depends on the method of solution in each of the subregions Ω_{ij}. For example, central difference approximations to equations (2.4) yield matrix equations

$$A_{ij}^{(k)} \bar{u}_{ij}^{(k)} = \bar{b}_{ij}^{(k)} \text{ in } \Omega_j \tag{3.1}$$

where each $A_{ij}^{(k)}$ is an $\eta_{ij}^{(k)} \times \eta_{ij}^{(k)}$ matrix and $\bar{b}_{ij}^{(k)}$ is a vector constructed from the values of f_{ij} on the grid, the boundary conditions of (2.1), and "pseudo-boundary" conditions furnished by the neighboring solutions. In this case, the vector $\bar{u}_{ij}^{(k)}$ are simply the solution of the matrix equations (2.5). Convergence of the Schwarz iterates (2.5) for this method can be found in [2], [3], [4], [5]. The numerical task for each of the processing units is the numerical inversion of the system in (2.5). Consequently, since the Schwarz process is inherently parallel, the computational time to obtain the numerical solution of (2.1) is the product of the number of Schwarz iterations with the sum of the solution time for the linear systems solvers and the communication time for the passage of pseudo-boundary dates between processors. Hence, a means for accelerating this process would be to reduce the number of Schwarz iterations needed for convergence and to reduce the computational time needed to provide the approximations $\bar{u}_{ij}^{(k)}$. One such means is to use the Fourier frequency techniques that have been so successful in accelerating the classical Gauss-Siedel iterative methods, e.g. multi-grid acceleration.

4. Fourier Frequency Acceleration

Fourier frequency accelerations owe their origin to observation made by A. Brandt in [1] of the fact that in an iterative method the magnitude of the high Fourier frequencies of the error is diminished considerably after only a few iterations. Since the number of frequencies present in the error is determined by the mesh size $1/\eta_k + 1$ of the grid G_k, low frequencies in one grid are high frequencies in a coarser grid. Hence, by successively changing the grid from fine to coarse one is able to achieve an acceleration of the original iteration. In this paper we will study the effects of the Schwarz iteration when the approximation $\bar{u}_{ij}^{(k)}$ are provided from grids G_k in (2.2) that sequentially changed from fine to coarse so as to capitalize on the maximal decrease of the high frequency components of the error. The parallel performance of these methods will be studied on an Alliant FX/8 multiprocessor. The actual computational experiments will be described in the final draft of this paper as the experiments are ongoing.

Bibliography

[1] A. Brandt, *Multi-level Adaptive Solutions to Boundary Value Problems,* Math. Comp., 31 (1977), pp. 333-390.

[2] L. Kang and D. Evans, *The Convergence Rate of the Schwarz Alternating Procedure (V) - for More Than Two Subdomains,* Intern. J. Computer Math., 23 (1987).

[3] G. Rodrigue, *Inner/Outer Iterations and Schwarz Algorithms,* Parallel Computing 2 (1985), pp. 205-218.

[4] G. Rodrigue and Y. Liu, *Convergence and Comparison Analysis of Some Numerical Schwarz Methods,* Lawrence Livermore National Laboratory Report, UCRL-98885, May 31, 1988.

[5] W.P. Tang, *Schwarz Splitting and Template Operators.* Ph.D. Dissertation, Computer Science, Stanford University, June 1987.

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48